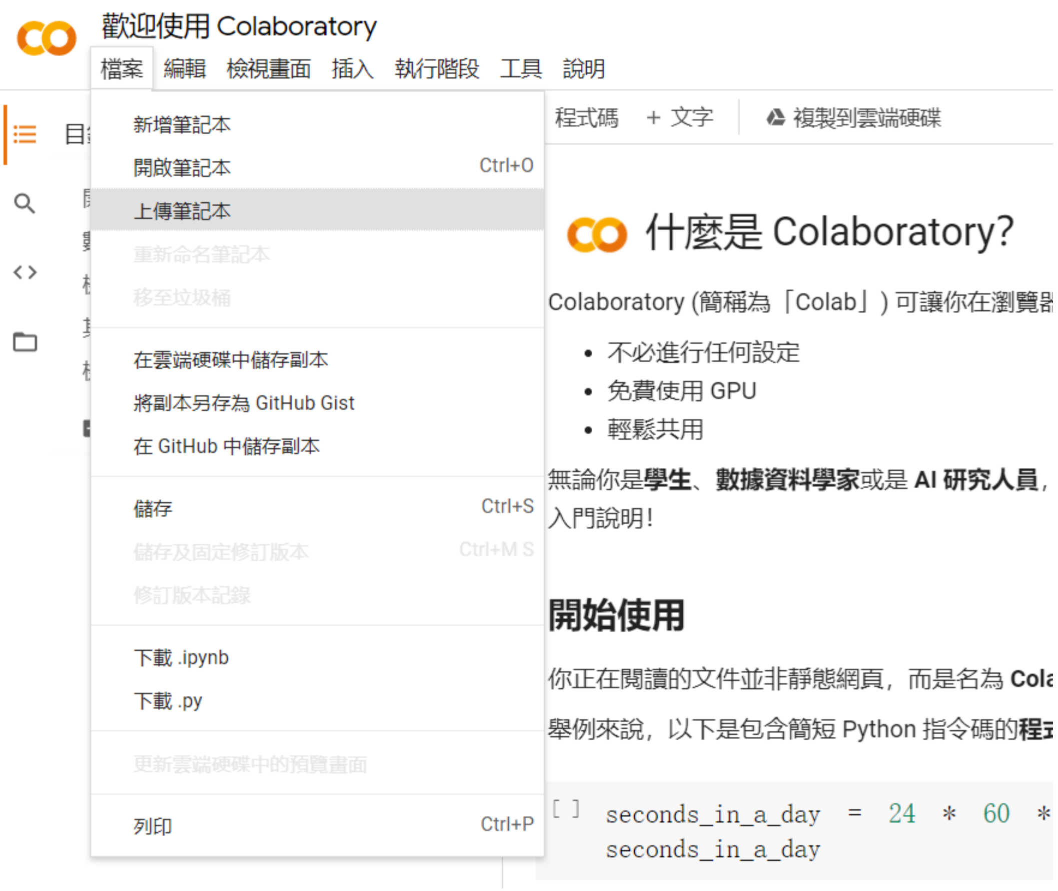
07360796 沈奕霖、07360194 陳維嶸

AI_Iris 模型預測_ReadMe

1. 上傳 AI_Project_Final.ipynb 到 Colab



什麼是 Colaboratory?

Colaboratory (簡稱為「Colab」) 可讓你在瀏覽器

- 不必進行任何設定
- 免費使用 GPU
- 輕鬆共用

無論你是**學生**、**數據資料學家**或是 **AI 研究人員**，
入門說明！

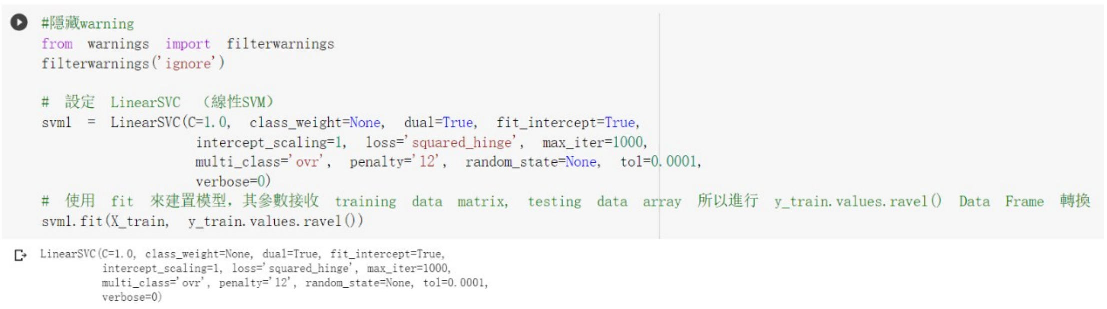
開始使用

你正在閱讀的文件並非靜態網頁，而是名為 **Cola**

舉例來說，以下是包含簡短 Python 指令碼的**程式**

```
seconds_in_a_day = 24 * 60 *
seconds_in_a_day
```

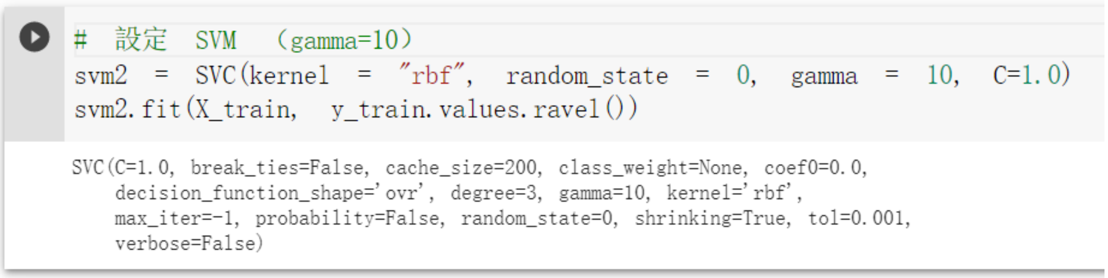
2. 線性 SVM



```
#隱藏warning
from warnings import filterwarnings
filterwarnings('ignore')

# 設定 LinearSVC (線性SVM)
svml = LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
                 intercept_scaling=1, loss='squared_hinge', max_iter=1000,
                 multi_class='ovr', penalty='12', random_state=None, tol=0.0001,
                 verbose=0)
# 使用 fit 來建置模型，其參數接收 training data matrix, testing data array 所以進行 y_train.values.ravel() Data Frame 轉換
svml.fit(X_train, y_train.values.ravel())
```

```
LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
          intercept_scaling=1, loss='squared_hinge', max_iter=1000,
          multi_class='ovr', penalty='12', random_state=None, tol=0.0001,
          verbose=0)
```

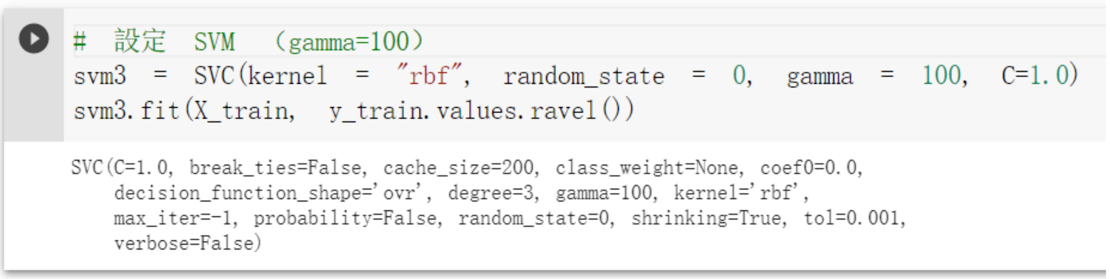
3. 非線性 SVM（gamma=10）



```
# 設定 SVM (gamma=10)
svm2 = SVC(kernel = "rbf", random_state = 0, gamma = 10, C=1.0)
svm2.fit(X_train, y_train.values.ravel())
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=10, kernel='rbf',
    max_iter=-1, probability=False, random_state=0, shrinking=True, tol=0.001,
    verbose=False)
```

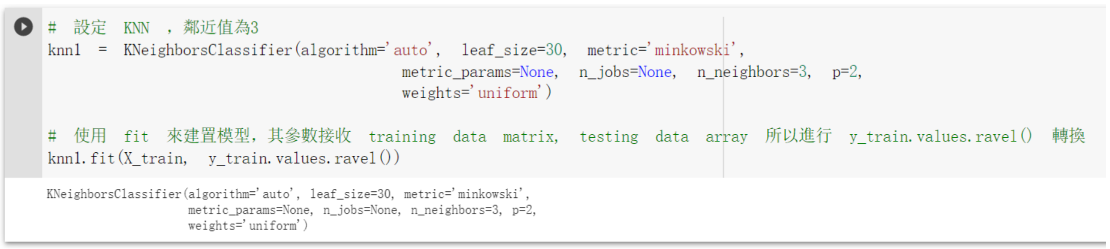
4. 非線性 SVM（gamma=100）



```
# 設定 SVM (gamma=100)
svm3 = SVC(kernel = "rbf", random_state = 0, gamma = 100, C=1.0)
svm3.fit(X_train, y_train.values.ravel())
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=100, kernel='rbf',
    max_iter=-1, probability=False, random_state=0, shrinking=True, tol=0.001,
    verbose=False)
```

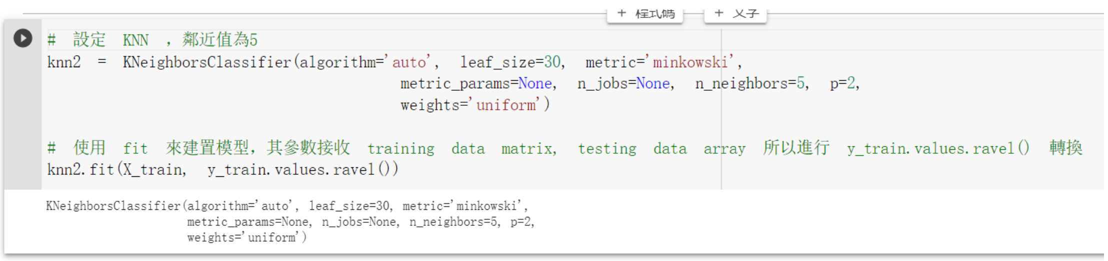
5. 鄰近值為 3 的 KNN



```
# 設定 KNN ，鄰近值為3
knn1 = KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                           metric_params=None, n_jobs=None, n_neighbors=3, p=2,
                           weights='uniform')

# 使用 fit 來建置模型，其參數接收 training data matrix, testing data array 所以進行 y_train.values.ravel() 轉換
knn1.fit(X_train, y_train.values.ravel())
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=3, p=2,
                     weights='uniform')
```

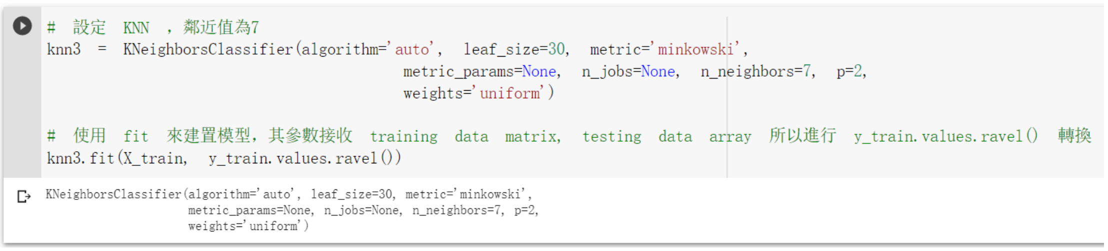
6. 鄰近值為 5 的 KNN



```
# 設定 KNN ，鄰近值為5
knn2 = KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                           metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                           weights='uniform')

# 使用 fit 來建置模型，其參數接收 training data matrix, testing data array 所以進行 y_train.values.ravel() 轉換
knn2.fit(X_train, y_train.values.ravel())
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
```

7. 鄰近值為 7 的 KNN



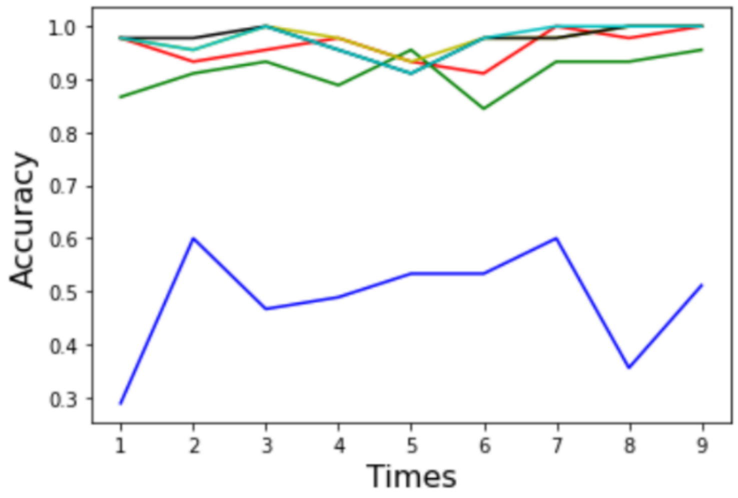
```
# 設定 KNN ，鄰近值為7
knn3 = KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                           metric_params=None, n_jobs=None, n_neighbors=7, p=2,
                           weights='uniform')

# 使用 fit 來建置模型，其參數接收 training data matrix, testing data array 所以進行 y_train.values.ravel() 轉換
knn3.fit(X_train, y_train.values.ravel())
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=7, p=2,
                     weights='uniform')
```

8. 執行和繪圖

```
[66]  #隱藏warning
      from  warnings  import  filterwarnings
      filterwarnings('ignore')

      #執行且繪圖
      loop_test()
```

```
Test time:10
SVM1平均準確度: 0.9577777777777777
SVM1最大準確度: 1.0
SVM1最小準確度: 0.9111111111111111

SVM2平均準確度: 0.9155555555555555
SVM2最大準確度: 0.9555555555555556
SVM2最小準確度: 0.8444444444444444

SVM3平均準確度: 0.49555555555555547
SVM3最大準確度: 0.6
SVM3最小準確度: 0.28888888888888886

KNN1平均準確度: 0.9733333333333334
KNN1最大準確度: 1.0
KNN1最小準確度: 0.9333333333333333

KNN2平均準確度: 0.9711111111111113
KNN2最大準確度: 1.0
KNN2最小準確度: 0.9111111111111111

KNN3平均準確度: 0.9733333333333334
KNN3最大準確度: 1.0
KNN3最小準確度: 0.9111111111111111
```


Accuracy
Times